Journal of Computer Science

Original Research Paper

Enhancing Indian Language Speech Recognition Systems with Language-Independent Phonetic Script: An Experimental Exploration

Jose Stephan and Muthayyan Kamalam Jayakumar

Department of Computer Science and Engineering, Noorul Islam Centre for Higher Education, Kumarakoil, Kanyakumari District, Tamilnadu, India

Article history
Received: 03-08-2024
Revised: 04-12-2024
Accepted: 31-01-2025

Corresponding Author:
Jose Stephan
Department of Computer Science and Engineering, Noorul Islam Centre for Higher Education, Kumarakoil, Kanyakumari District, Tamilnadu, India
Email: jose.stephen.05@gmail.com

Abstract: India has a distinct linguistic profile that makes it extremely challenging to train Automatic Speech Recognition (ASR) systems correctly because most Indian languages have limited training material and are generally low-resource in nature. However, due to their similarity in phonemes, these languages offer an opportunity to create a single speech recognition technology. This study proposes an optimized phonetic script which can be generalized for all major Indian languages. To prove the efficiency of the phonetic script, speech recognition models using phonetic and language scripts in Hindi and Malayalam language were created using a wave2Vec2-based Deep Neural Network (DNN) model via transfer learning. Furthermore, a model based on Long Short-Term Memory (LSTM) is created to translate phonetic script text back into its original languages. The findings show that the phonetic script ASR model performed noticeably better than the language-specific model, reducing WER roughly to 2%, especially for the Hindi language, which is further reduced up to 1% for the model trained with mixed language. This demonstrates the model's ability to improve performance by using cross-lingual phonetic similarities. This study establishes the foundation for cross-linguistic, scalable ASR systems that use phonetic similarities to enhance ASR performance in low-resource language contexts across India.

Keywords: Deep Neural Network, End-to-End Automatic Speech Recognition, Language Independent Pronunciation Script, Speech Recognition for Indian Languages, Transformer-based Wave2Vec2

Introduction

The most common communication concept in people's social bonding is speech, and as human beings, information sharing can easily be done through this. Scientists and engineers have been amazed at the convergence of the possibility of speaking naturally and reacting correspondingly to spoken languages during the past century. The result of this is that machines have been created to replicate human functions. The problem of automatic speech recognition has been approached progressively since the 1930s when Homer Dudley of Bell Laboratories proposed a system model for speech analysis and synthesis. Dudley (1939) went from a basic machine that responds to a limited set of sounds to a sophisticated system that responds to naturally spoken language and takes into account the varying statistics of the language in which the speech is produced.

India is home to many native languages, and it is also common for people to speak and understand more than one language or dialect, which can entail the use of different scripts as well. There are 22 official languages, and 121 languages are spoken as mother tongues, which is defined as the first language a person learns. Due to the limited resources available, developing Automated Speech Recognition (ASR) systems for Indian languages is a difficult task. The act of switching between different languages during a single discussion is known as code-switching. One of the most often utilized languages for code-switching is English. Multilingual and code-switching ASR for Indian languages are the two main objectives of the Indic Diwan (2021); Chen *et al.* (2018).

The fact that different Indian languages have diverse grapheme representations presents a significant challenge when developing ASR systems for them in multilingual environments. Indian languages have the same phonetic foundation but distinct writing systems. This led to the development of a uniform label set for n Text-to-Speech (TTS) synthesis in 13 Indian languages Pratap *et al*. (2020). To translate grapheme-based Indian language text to phoneme-based CLS, a rule-based uniform parser for 13 Indian languages. Baby *et al*. (2016) CLS has demonstrated efficacy for TTS systems created with contemporary end-to-end frameworks as well as Hidden Markov Model (HMM) based techniques Prakash *et al*. (2019). The majority of Indian languages are either Dravidian or Aryan or a combination of the two. As part of the investigation, two major Indian languages, Hindi and Malayalam, from the north and south were selected. Hindi belongs to the Indo-European family, and Malayalam belongs to the Dravidian family. There are numerous phonetic similarities among Indian languages, which raises the prospect of a small, shared phone set across all of them Shahnawazuddin *et al*. (2017).

In the work by Toshniwal *et al*. (2018), a single sequence-to-sequence (ASR) Automatic Speech Recognition model was designed and trained on a variety of nine Indian languages, in which the script overlap was minimal Jinal and Dipti (2016). They used the data from each language separately to link all languages in joint training of a grapheme sequence to the sequence model with the union of all grape-meme sets. When unsupervised, the model, which by itself captures no explicit information related to language identity, gets better than comparable sequence-to-sequence models trained on every single language by 21% in recognition performance. Furthermore, they increased the performance by an extra 7% relative and abolished the linguistic uncertainty with an extension to a language identification system to the model features as additional input features.

Kumar *et al*. (2021) introduced the training of multilingual and code-switching ASR for Indian languages with an internal rule-based CLS representation at the level of phoneme Klein *et al*. (2017). The two systems they proposed were the E2E ASR and the complex contextual understanding. During the first system, the native language script can be recovered by applying data-driven back-end training on the E2E model on the CLS representation. In this second method, we modify the E2E model by allowing training to be performed at the same time using (CLS) representations and the original characters from the native language. They presented the outcomes of the Indic ASR Challenge 2021 that carried on the multilingual and code-switching tasks.

A single multilingual speech recognition system that can recognize any of the training languages was constructed in this research led by Kim and Seltzer (2018) through recent advancements in end-to-end voice recognition. Huggingface (2024a) One of the main steps they took towards achieving this goal was introducing a ubiquitous character set that serves all languages. Moreover, they constructed a language-specific gating system that is capable of evoking representations in an internal mode that is tuned to the language. The Microsoft Cortana task was apt to demonstrate what we propose, and the results obtained indicate that their system overtakes both a standalone monolingual system and a full-fledged system that uses a multitask learning strategy. The authors then showed how this framework could be used to create either a bilingual model for code-switching or a monolingual one with speech recognizers as the initial ones.

Anirudh Gupta and his colleagues have developed a CLSRIL-23 audio pre-trained model based on the use of self-supervised learning on 23 Indic languages from the raw audio data. It is uprighted on wav2vec 2. With the onset of language, humans innately transfer all language's quantized latent and share them simultaneously. The problem is tackled by training a contrastive task across latent voice representations that have been masked Gupta *et al*. (2021). To examine the impact of monolingual and multilingual pretraining, they compared the language-wise loss during pretraining. Using speech representations that encode phonetic similarity between languages, their research demonstrates that multilingual pretraining outperforms monolingual training in downstream task performance as well as in the acquisition of such representations. Performance on various downstream speech recognition finetuning tasks was also compared. When a multilingual pre-trained model is employed for finetuning in Hindi, a 5% drop in WER and a 9.5% drop in CER were seen. Additionally, all of the code models have open sourcing. To aid in the study of speech recognition for Indic languages, the model CLSRIL-23 was trained on 23 languages and around 10,000 h of audio data. The authors anticipate that, especially for Indic languages with limited resources, cutting-edge systems will be developed utilizing the self-supervised technique.

Malek *et al*. (2017) looked at an ASR system with background music Lee *et al*. (2018). They took two different routes. The first method relied on training the acoustic models under multiple conditions. The second used preprocessed data for acoustic model training after denoising autoencoders. The findings demonstrated that every method they looked into might considerably increase the ability to identify speech that has been altered by music. The researchers presented a front-end speech parameterization method that is resistant to both noise and pitch fluctuations. Pradhan *et al*. (2015) Children's clean and noisy speech was used for testing, and speech data from adult and kid speakers was used to train the ASR system. The objective was to improve the ASR system's resistance to noise. That method's efficacy has been

confirmed on an ASR system created using acoustic modelling based on DNN-HMM.

A new method with several properties, including format frequencies, energy measures and zero crossing rates, was employed by some writers in Jinal and Dipti (2016) state that the Hidden Markov Model Toolkit is used for measuring the performance and various error characteristics in the Gujarati language Jinal and Dipti (2016). Some have employed the Hidden Markov model with Mel frequency cepstral coefficients. Acoustic models-based tri-phones specifically built for the Punjabi language have been employed for continuous speaking.

Scientists have investigated other languages, like Thai, Romanian, Hungarian, Chinese, Japanese, and Indian. At the same time, others are focused on building language recognition systems for different languages using document text online. The architecture of the speech recognition system for this language, constructed on the basis of elements from the Hidden Markov Modeling Toolkit (HTK), has been highlighted by Burileanu *et al.* (2010). It has two components: It will go through the process of acting as well as decoding.

The multilingual transformer-based architecture of the wav2vec 2.0 deep neural network offers significant advantages in handling multiple languages, particularly for Automatic Speech Recognition tasks. A Conneaushows that self-supervised learning approaches like wav2vec 2.0 enable shared representations, significantly improving multilingual ASR performance without additional task-specific annotations. Conneau *et al*. (2019); Babu *et al*. (2022) demonstrate the power of the framework in multilingual pretraining for cross-lingual transfer in speech recognition tasks. Babu *et al*. (2022); Pratap *et al*. (2020) in his work highlight how multilingual architectures effectively handle code-switching scenarios. Prakash and Murthy (2020), from the literature, understand that Wav2Vec 2.0 achieves state-of-the-art results on high-resource datasets and performs significantly better on low-resource languages due to its self-supervised pretraining and finetuning capabilities.

The motivation for carrying out studies on Indian language Automatic Speech Recognition (ASR) systems with a Language-Independent Pronunciation Script (LIPS) arises from this urgency that needs to be addressed, which involves the difficulties of multilingualism in India. However, because of the numerous languages distributed throughout the country, each presenting peculiar phonetic characteristics and scripts, traditional ASR systems are unable to correctly interpret diversity. The purpose in this regard is to investigate the possibility of a universal method of language-independent sound representation that supersedes individual languages and scripts, with the end result being a unified solution for application in ASR speech recognition for any linguistic basis. The study intends to face these issues and establish the road of adaptable and inclusive ASR, which competently deals with all the complexity-code mixing, distinct scripts and linguistic variations in the Indian scenario. The research aims to provide enriching insights for the future of whole language-agnostic ASR technologies that can support many Indian languages by considering the peculiarities of the Indian linguistic landscape.

Materials and Methods

Materials

This subsection describes the speech corpora and Text data used in the study.

Speech Corpora

For Hindi language custom dataset is prepared from collecting data from various sources like Common Voice, Open SLR together with in-house prepared data. The number of hours of audio is about 8 hours and the sampling ratio is 16 kHz.

For Malayalam language custom dataset is prepared from various sources like Common Voice, SMC Malayalam Speech Corpus together with in-house prepared data. The volume of audio data is about 10 h and the sampling rate is 16 kHz.

Text Data

Text data is used for creating machine learning model for pronunciation script to language script conversion. For both Hindi and Malayalam language it is collected from Open-Speech-EkStep (2022) site, https://github.com/Open-Speech-EkStep/vakyansh-models#pretrained-asr-models.

Equipment

This subsection describes the hardware and software used. Hardware resources used include GPU: NVIDIA A4000 (16GB), CPU: Intel I7 8700, RAM: 32 GB, Storage: 500 GB SSD. Software Frameworks include Deep Learning Libraries: PyTorch, ASR Toolkit: Fairseq-S2T, Feature Extraction: Librosa / torchaudio, Training Pipeline: Hugging Face Transformers / Wav2Vec2.

Methods

The full process of constructing a speech recognition system uses the approach, including data collection, preparation, the selection of a pretrained model and finetuning before making a rigorous evaluation. In a similar way, each step of the methodology is drawn in mind to face the disadvantages that are raised by verbal diversity, which is an integer part of India, with the overall objective of increasing the adaptability quality as well as the performance of ASR systems.

A baseline E2E speech recognition model was created for two languages. The basic block diagram of the

procedure followed in building the baseline system is given in Fig. (1).

Data Collection

Formulation of a data set is the first and very important step in building a model of a speech recognition system.

Table (1) demonstrates that in 13% of the studies, the authors attempted to develop a recorded dataset of their own to verify their methods and employ them in their experiments. Subsequently, the CHiME and REVERB Challenge database, the TIMIT database, Aurora, LibriSpeech, SWB2000 and AMI were the most widely used datasets. These made up 41 % of the papers. A further 12% did not use any dataset, 11% did not specify which dataset they used, and they described their methodology without providing any validations. A few of the datasets were only ever used once. They made up 8% of the papers and were categorized as "other." It is advised that researchers in the future either generate their own ASR data or use pre-existing data sets (CHiME and REVERB Challenge database, the TIMIT database, Aurora, LibriSpeech, SWB 2000 and AMI) in light of the explanations provided in this study.

In this study, we partially created the data for building the model, and the rest was collected from freely available sources. The first step for creating a speech corpus is to formulate the text to be recorded. For the Hindi language, we collected the summary reports entered by the call dispatcher from emergency control rooms of different Hindi-speaking states. The collected texts were manually analyzed and corrected with the help of language experts, and the texts were selected based on phonetic coverage. After creating the text corpus, the next step was to record and collect speech from the native speakers. For that, the proposed method built a web-based recording interface in which any user with an internet facility can sign up and log in to record the speech Anand *et al*. (2012). The collected data is mixed with data collected from the open-source domain, and together, a speech corpus is built to build the Hindi language model. For the Malayalam language, this research formulated the corpus based on maximum triphone coverage. The speech data was collected in a normal room from around 80 persons. Mihajlik *et al*. (2010) The collected speech data is mixed with openly available datasets such as Mozilla Common Voice and SMC Malayalam Speech Corpus dataset. The collected speech data for the Hindi language is around five hours, and for Malayalam, it is around six hours. Since the formulated speech data shall be used for finetuning the pre-trained model Cross-Lingual Speech Representations for Indic Languages (CLSRIL-23), which is trained over 10,000 h for 23 Indian languages, the data shall be more or less sufficient to cover distinct phonetic characteristics for these languages.

Fig. 1: Workflow of the proposed study

Table 1: Applied datasets in the reviewed papers

Dataset	Number of papers	Percentage (%)
Real speech datasets	15	13
Not mentioned	12	11
none	13	12
CHiME and Reverb Challenge database	12	11
TIMIT database	9	8
Videos dataset	8	7
Aurora	7	6
Libri Speech	7	6
SWB2000	6	5
AMI dataset	5	4
Language dataset	4	4
Google dataset	2	2
Various corpus	3	3
Others	9	8
Total	112	100

Data Preparation

Before utilizing the collected speech and text data for model construction, preprocessing is imperative. The Malayalam language has one main challenge: Multiple representations of the same characters, such as chilli (individual character or combination of base character with Chandrakala and zero width joiner) and the vowel character AU. To address this issue, normalization procedures were applied to the collected text data. Data normalization was done programmatically using self-developed Python scripts. The primary outcomes of the data preparation stage include the creation of a lexicon file, which organizes words in the data into their constituent characters and a dictionary file. The dictionary file captures the characters present in the data along with their respective frequencies, providing essential insights for the subsequent stages of model building. Kim and Seltzer (2018) state that this preprocessing step ensures that the data is appropriately structured and ready for effective utilization in the model development process.

Selection of Pre-Trained Model and Finetuning

Employing transfer learning to construct baseline models necessitated the utilization of a pre-trained model, specifically one trained with 10000 h of Indian language data, publicly accessible in the open domain. Prakash *et al*. (2019) Creating g "codebooks," each with v entries of quantized representations and selecting a quantized representation ε from each codebook to resemble a matching portion of the latent vector z is the basic process of Wav2vec2 pretraining. The product vector quantization

process is finished when all *e* are concatenated. Discrete objects are selected in a way that is differentiable from codebooks using the Gumbel softmax. For the entries in various codebooks, the *z* from the feature encoder output is transferred to l $\in R\ G \times V$ logits. Next, Eq. (1) gives the probability of selecting the v^{-th} entry in codebook *g*, where $n = -log\ (-log\ (u))$, τ is a non-negative temperature, and *u* are uniform samples from the continuous uniform distribution *U* (0, 1). As a result, in a forward pass, i = argmaxj (P_g, j) and in a backward pass, the Gumbel softmax gradient is encountered:

$$P_{g,v} = \frac{exp(l_{g,v}, n_v)/\tau}{\varepsilon_{k=1}^{v}\ exp\ (l_{g,v,} + n_v)/\tau} \quad (1)$$

The goal of training is to reduce the amount of loss. *L* is equal to $L_m + \alpha\ L_d$, where α is a tunable hyperparameter, L_d is a codebook diversity loss, and L_m is a contrastive loss. Equation (2) defines the contrastive loss. This equation represents the context network output at a masked time step *t*, and $_{it}$ is the real quantized latent that the model must distinguish from *K* quantized distractors Q_t. The cosine similarity between the context representation and various quantized speech representations is known as the *sim* (*a*, *b*):

$$L_m = -log \frac{exp\left(\frac{sim(c_t, q_t)}{K}\right)}{\sum_{\tilde{q} \sim Q_t}\ exp\left(\frac{sim(c_t, \tilde{q})}{K}\right)} \quad (2)$$

The diversity loss L_d, the second loss function, incentivizes the model to employ the codebook entries more fairly. By "maximizing the entropy of the average softmax distribution *l* over the codebook entries for each codebook $\overline{P}g$," Babu *et al*. (2022) created the diversity loss. Thus, Eq. (3) is the final equation for L_d. Keep in mind that *V* represents entries in the codebooks, and *G* stands for codebooks:

$$L_d = \frac{1}{GV} \sum_{g=1}^{G} \sum_{v=1}^{V} \bar{p}_{g,v}\ log \bar{p}_{g,v} \quad (3)$$

To optimize this pre-trained model, the configuration file from the pre-trained wav2vec2 base model on Librispeech was employed, extracting hyperparameter values for the subsequent finetuning process. Anand *et al*. (2012), during finetuning, the pre-trained model underwent training on the collected dataset, adapting it to our needs. The speech data gathered was partitioned into three sets, with approximately 10% allocated for testing, 5% for validation and the remaining 85% for training. The training process, encompassing about four hours of data for Hindi and around five hours of data for Malayalam, achieved convergence at around 140 epochs for Hindi and 290 for Malayalam. Given this training set is less than 10 h, we adhered to the basic hyperparameter settings provided in the fairseq example for such scenarios.

Formulation of Pronunciation Script (PS)

The next step was to create the pronunciation script and create a speech recognition model using it. Pronunciation Script was designed keeping in mind with the following:

- Shall contain all sounds in the two languages
- Shall utilize the pronunciation similarity of Indian languages
- It shall be expandable so that it handles the pronunciation of all major Indian languages
- Similar-sounding characters shall have only one representation
- Only a single letter shall be used to represent a sound

The sample scripts are given in Fig. (2).

The expected advantages of using pronunciation script in modelling are as follows:

- The Language-independent property of PS enables the training of the model using multiple Indian languages, which helps to build an independent speech recognition model
- Output characters will be reduced, which shall help to get good results with low data

The expected challenge in using PS in speech recognition modelling is that we need to handle language-dependent pronunciation rules for each language because the pronunciation model will be language-dependent.

Training Model with Pronunciation Script

The training procedure for the model closely mirrors that of the baseline model shown in Fig. (3), with the addition of a crucial step: Language script to Pronunciation Script (PS) conversion. In this step, the training text corpus for both Hindi and Malayalam transforms PS using a pronunciation model. The conversion process employs a pronunciation rule parser designed to recognize the language within the language script. Subsequently, language-dependent pronunciation rules are applied to facilitate the conversion from language script to PS. This essential augmentation in the training pipeline ensures that the model effectively incorporates phonetic nuances inherent in the linguistic diversity of Hindi and Malayalam. This step of training text corpus transforms PS for both Hindi and Malayalam languages using a pronunciation model, which is created by considering parameters like phoneme inventory, stress and intonation.

The converter uses a rule-based parser that is intended to capture the various languages in the scripts that are similar to the mother language. Thus, language-based speech pronunciation rules are included in the training pipeline to minimize this conversion into PS-script. This essential augmentation in the training pipeline is important, as it introduces the phonetic detail of the linguistic diversity of Hindi and Malayalam in the model.

DOI: 10.3844/jcssp.2025.1176.1186

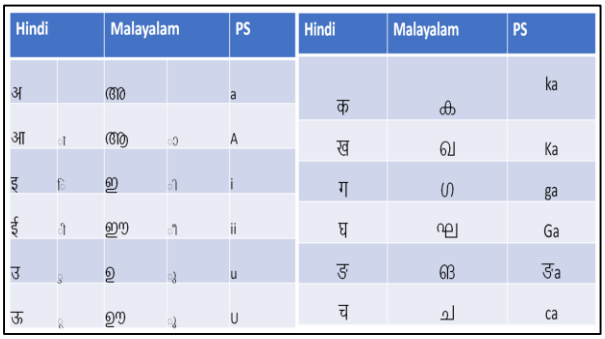

Fig. 2: Sample scripts in Hindi, Malayalam and Pronunciation Script (PS)

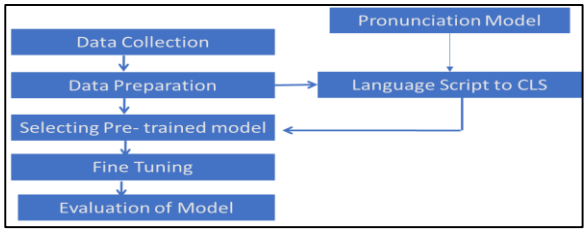

Fig. 3: Flow diagram for the training phase

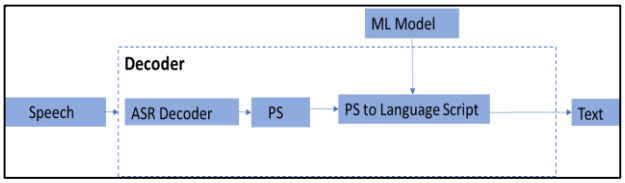

Fig. 4: Decoder of the proposed work

Table 2: Confusions in PS to native script mapping

Pronunciation script	Possible mapping
kapda	कप्ड़ा, कपड़ा
bachpan	बचपन,बच्पन

The pronunciation rules that I followed when speaking the Hindi language were the result of the discussions with Hindi experts and the use of online sources. Malek *et al*. (2017) state that experts of language first assign the appropriate rules to the topics under analysis and then verify these rules by considering the corpus of original sentences. After being validated, the rules were then implemented and used to convert the written language to the oral script, and training on how to apply the scripts was conducted by using the transformed scripts.

For the Malayalam language, pronunciation rules were adapted from (Mihajlik *et al*., 2010). Following the conversion of the text corpus to Pronunciation Script (PS) based on these rules, the model was trained using the same methodology as the creation of the baseline system.

Post-training, the model generates outputs in the Pronunciation Script (PS), which are then converted back to the Language Script (LS) using a machine learning model. This conversion step ensures coherence between the model's predictions and the original language script, finalizing the comprehensive training process.

Decoding

The decoding module block is shown in Fig. (4). This figure clearly shows the end-to-end architecture because it processes raw audio waveforms directly and produces meaningful representations or outputs without requiring extensive intermediate steps or manually designed features. The raw speech provided as the input is converted to recognized text with the help of ASR Decoder, which converts raw speech to pronunciation script, then text-to-text conversion of pronunciation script to the required language script will be done by pronunciation script to language script block. However, due to the absence of a direct one-to-one mapping for language script to Pronunciation Script (PS) conversion, the reverse transformation of PS to the language script poses a significant challenge. Table (2) provides illustrative examples of such complexities. Given the inherent difficulties in retrieving the language script directly from the PS using a one-to-one mapping, a Machine Learning (ML) model was employed for this conversion.

The Open Neural Machine Translation (ONMT) toolkit served as the basis for creating neural machine models designed specifically for the conversion of PS to the Language script. Kim and Seltzer (2018) In the training data, each character was separated by inserting a space character and word delimiters were marked by a pipe character before training using the default configuration. The default architecture consists of a 2-layer LSTM with 500 hidden units on both the encoder and decoder Nayar (1980). The ML model is then integrated with the ASR system, and then. Subsequently, the model's efficacy is evaluated using the same test set employed to test the baseline system. This approach facilitated a nuanced and context-aware conversion from the pronunciation script to the original language script.

Results

The results of the speech recognition model will be evaluated based on key performance metrics, including word error rate, character error rate and phoneme recognition accuracy. Because both Word Error Rate (WER) and Character Error Rate (CER) are connected with text ambiguity, it is a regularly used metric to compare speech recognition model performance. Lower numbers indicate lesser discrepancies or better outcomes. The word error rate is a measure of the word-by-word differences between two texts. As a result, even when trained with distinct loss functions, different models can be compared. Equation (4), where *S* denotes the number of substitutions, *D* the number of deletions, *I* the number

of insertions and *N* the total number of words, defines *WER*:

$$WER = \frac{\text{S+D+I}}{\text{N}} \quad (4)$$

CER, similarly, evaluates error at the character level, providing a finer assessment of the model in Eq. (5):

$$CER = \frac{S+D+I}{N} \quad (5)$$

where, *S*, *D* and *I* denote substitutions, deletions and insertions at characters.

As part of the study, speech recognition models were created based on a language script, pronunciation script, and a combination of both Hindi and Malayalam languages using a pronunciation script. Table (3) Figs. (5-6) show the comparison of these models with other models available in the open-source domain. GitHub (2024) for Hindi language, the Wav2Vec2-XLS-R-300M model gives WER 32.5 and CER 8.75; for SakshiRathi77/wav2vec2-large-xlsr-300m-hi-kagglex which is a variant of wav2vec2-large-xlsr-300 which is available in hugging face gives WER 55.96 and CER 19.12. The whisper-based model gives WER 13.42 and CER 5.68. For SakshiRathi77/whisper-Hindi-kagglex, a variant of the whisphermodel gives WER 23.14 and CER 10.44. An open-source model, Vakyansh open-source Models, was tested with the same test data, and our models gave 29.33 as WER and 15.24 as CER. The referenced paper states that WER for the Baseline E2E Model, CLS E2E Model and Dual Script E2E model are 26.5, 26.2, and 25.9, respectively. The WER and CER for the experimental model were created based on language scripts and pronunciation scripts using the Wave2Vec2 framework, which gave 21.77, 8.35, 19.8 and 7.1, respectively. Hindi- Malayalam mixed model based on pronunciation script gives a WER of 18.71 and CER of 6.77, which is the best result among the created models.

We also created models based on language and pronunciation scripts for the Malayalam language. At the initial stage of the study, the WER and CER were high, but it was reduced after optimizing the training dataset by mixing up different open-source datasets such as Mozilla Common Voice, SMC Malayalam Speech Corpus dataset together with self-created dataset and then by filtering out very short sentences and word data from the dataset. The created model is then tested with created test data consisting of around 2000 sentences. The model built with the combination of Malayalam and Hindi was also tested with the Malayalam test data. The results obtained compared with other models available in the open-source domain are provided above. Hirayama *et al*. (2015; Huggingface (2024b) stated that when Mozilla Common Voice 11.0 is used as test data, the Malwhisper-v1-medium model gives WER and CER as 61.84 and 16.41 and 70.49, 17 as WER and CER when SMC Malayalam Speech Corpus dataset is used a test dataset. Vakyansh's open-source model was tested, and our test data gives 68.33 as WER and 15.74 as CER. The WER and CER for the experimental model were created based on language scripts and pronunciation scripts using the Wave2Vec2 framework, which gave 13.43, 1.744, and 14.42, 1.54, respectively. The Hindi-Malayalam mixed model based on pronunciation script gives a WER of 15.26 and a CER of 1.6. The average sentence error of the Malayalam-Hindi mixed model is 58.31% for Malayalam and 88.05% for Hindi. The response time for the model is about 40 sentences per second for both languages.

The test data for evaluating the models was prepared by mixing various openly available datasets, and it was deliberately made different from the training and validation data. For Hindi, the test set consisted of 319 sentences, and for Malayalam, it consisted of about 2000 sentences.

Table 3: Experimental result and comparison with other models

		Evaluation-Results	
Language	Methods	WER (%)	CER (%)
Hindi	Wav2Vec2-XLS-R-300M	32.50	8.75
	SakshiRathi77/wav2vec2-large-xlsr-300m-hi-kagglex	55.96	19.12
	Whisper	13.42	5.68
	SakshiRathi77/whisper-hindi-kagglex	23.14	10.44
	Open-source Vykyash model	29.33	15.24
	Baseline E2E model	26.50	
	CLS E2E model	26.20	
	Dual Script E2E model	25.90	
	Wav2Vec2 with LS	21.77	8.35
	Wav2Vec2 with PS	19.8	7.10
	Wav2Vec2 with Hin-Mal combined LS	18.71	6.77
Malayalam	Malwhisper-v1-medium with Mozilla Common Voice 11.0	61.84	15.41
	Malwhisper-v1-medium with SMC Malayalam speech corpus dataset	70.49	17.00
	Open-source Vykyash model	68.33	16.74
	Whisper medium Malayalam	38.62	7.33
	Wav2Vec2 with LS	13.43	1.744
	Wav2Vec2 with PS	14.42	1.54
	Wav2Vec2 with Hin-Mal combined LS	15.26	1.60

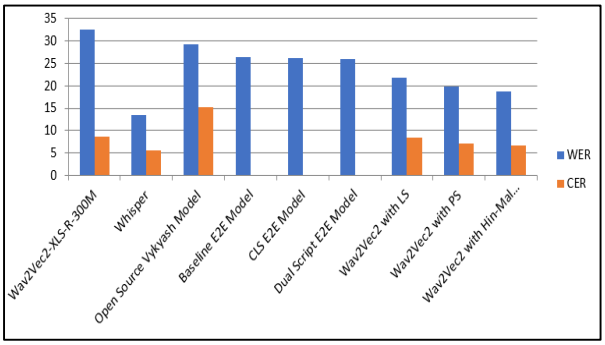

Fig. 5: Experiment result of Hindi LS and PS and comparison with other models

Discussion

Table (4) provides valuable insights into the accuracy and perplexity of the PS to native language script conversion for Hindi, employing the machine learning model and the graphical representation is depicted in Fig (6). Accuracy measures the precision of the conversion process, showcasing how well the system aligns phonetic representations with the actual language script. A higher accuracy indicates a more faithful conversion. Simultaneously, perplexity quantifies the uncertainty or complexity of the language model in predicting the native script from the pronunciation script. The accuracy and perplexity of the model in each step is given in Fig. (7).

Table 4: Machine learning model accuracy of pronunciation script to native language script conversion (Hindi)

Steps	Accuracy	Perplexity	Time(sec)
1000	51.0714	5.77952	720
2000	99.0069	1.03945	700
3000	99.3851	1.02363	750
4000	99.5483	1.01826	723
5000	99.5431	1.01713	680
6000	99.5457	1.01723	693
7000	99.6308	1.01289	711
8000	99.6451	1.013	726
9000	99.6484	1.01196	611
10000	99.5977	1.01365	623
11000	99.6484	1.01074	734
12000	99.6952	1.01072	684
13000	99.662	1.01143	756
14000	99.6802	1.01133	677
15000	99.6939	1.01063	698
16000	99.6997	1.0103	703
17000	99.6952	1.01034	674
18000	99.7413	1.00933	713
19000	99.753	1.0088	739
20000	99.7348	1.00858	716
21000	99.7296	1.00901	684
22000	99.7355	1.00847	749
23000	99.6581	1.01052	765
24000	99.7374	1.00898	757
25000	99.7017	1.0091	698

A lower perplexity suggests a more confident and efficient conversion process. These metrics collectively serve as crucial indicators of the effectiveness and reliability of the proposed conversion approach, shedding light on the model's performance in accurately transcribing phonetic representations into the native language script. The training set consists of around one lakh sentences, and the validation set is around 5000 sentences. The accuracy shown in the table is based on the validation data. The model created at step19000 was used in PS for native language script conversion.

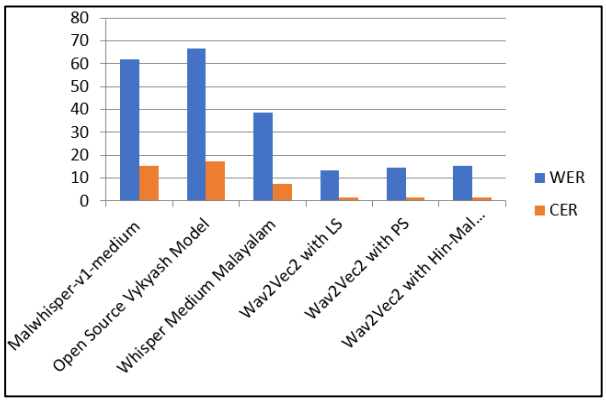

Fig. 6: Experiment result of Malayalam LS and PS and comparison with other models

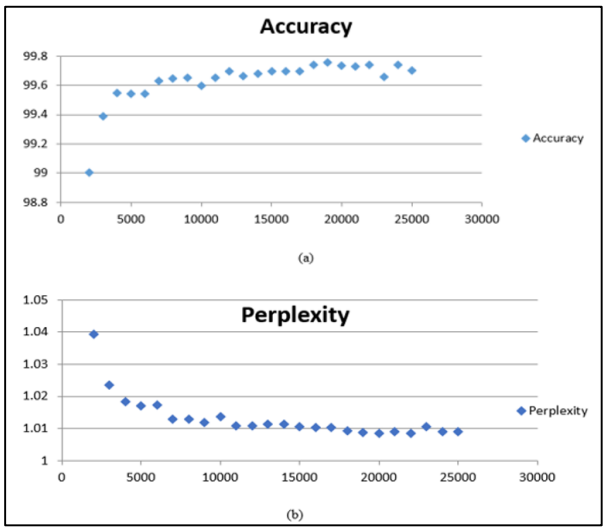

Fig. 7: Performance of the machine learning model for = PS to native language script conversion

The perplexity is calculated as follows.

For a language model predicting a sequence $W = (w_1, w_2, w_N)$, the perplexity is calculated as an equation (Continuumlabs, 2024):

$$PPL(W) = e^{-\frac{1}{N}\sum_{i-1}^{N} \ln P(w_i \mid w_{1:i-1})} \quad (6)$$

where:

- N: Total number of tokens (e.g., words or phonemes) in the sequence W
- $P(w_i|w_{1:i-1})$: The conditional probability of the i^{th} token given its context (the preceding tokens)

Accuracy is calculated as follows in Eq. (7).

$$Accuracy = 100 - WER \quad (7)$$

where, $WER = \frac{S+D+I}{N}$.

Conclusion

Here, the work is done on the complexities of the languages of India. ASR is considered a base principle. We have evolved an optimized phoneme script with the ability to accurately render these assorted language sounds within the Indian context. With the integration of a grammar-based parser, the adaptation between words written in one script language and phonetic alphabets and vice versa was made possible. The parser was equipped with language-dependent rules of pronunciation, which in turn built a base structure for more efficient language processing. It was our ground-level ASR system that showed alluring results based on our transformer Wave2Vec deep neural network, working with both Hindi and Malayalam for the core language. Through language script modelling, the rate of improvement was 4-5%, scored higher than the recognized sector. Moreover, this can be improved through the shorting of the different languages and mixing, and it can be adopted as the foundation for a successful multilingual ASR system. This shows that our approach is indeed well-suited to the task of audio pattern recognition in India and its ability to deal with the rich and varied phonetic characters and phonetic features of Indian languages. The multi-faceted design and evaluation of our system of Automatic Speech Recognition brings about insights that will guide us towards reaping the benefits of natural phonetic profiling and the use of ordinary sounds in Indian languages. Our model not only improves accuracy but also suggests an idea that would lead to the development of speech recognition technology customized to the speech patterns of people belonging to different regional and local languages of India. As our work on this front continues to evolve and expand, the chance that the broader field of LM ASR research will benefit from our findings remains open-ended. As just a set of diverse symbols, PS allows multilingual speech models to be developed that can be employed to recognize different languages. The second would focus on increasing the accuracy of the speech recognition model by checking and assessing the use of multiple languages. The potential application of this study is that it can be a building block to build a uniform model that can recognize multiple Indian languages, enabling the creation of multilingual support speech recognition systems for Indian languages.

Future Research and Gaps

The possible future developments of this study on ASR in India focus on more refinements to phoneme scripts and the use of grammar for India's various languages from potential parser developers that will improve the precision and speed of language processing for India's multilingual society. It has been ascertained that the transformer Wave2Vec DNN can be enhanced further, and these enhancements can be translated into higher efficiency for multilingual ASR systems. This method can greatly contribute to the advancement of personalized speech recognition for many of the regional and local languages in India as it makes continuity and styling of merging different languages with other simpler ones, as well as the improvement of phonetic profiling of these languages quite uncomplicated. The expansion of this research can also help contribute to progress within the ASR subfield.

Acknowledgment

The Department of Computer Science and Engineering at the Noorul Islam Centre for Higher Education is gratefully acknowledged by the authors for their cooperation with this undertaking. Author 1 and Author 2 contributed equally to this study.

Funding Information

The authors have no support or funding to report.

Author's Contributions

Jose Stephan: Conceptualization, methodology, Investigation, data curation, writing-original draft preparation.

Muthayyan Kamalam Jayakumar: Editing, supervision, project administration.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues involved.

References

Anand, A. V., Shobana Devi, P., Stephen, J., & Bhadran, V. K. (2012). Malayalam Speech Recognition System and its Application for Visually Impaired People. *2012 Annual IEEE India Conference (INDICON)*, 619–624. https://doi.org/10.1109/indcon.2012.6420692

Babu, A., Wang, C., Tjandra, A., Lakhotia, K., Xu, Q., Goyal, N., Singh, K., von Platen, P., Saraf, Y., Pino, J., Baevski, A., Conneau, A., & Auli, M. (2022). XLS-R: Self-supervised Cross-lingual Speech Representation Learning at Scale. *Interspeech 2022*, 2278–2282. https://doi.org/10.21437/interspeech.2022-143

Baby, A., N.L., N., Thomas, A. L., & Murthy, H. A. (2016). *A Unified Parser for Developing Indian Language Text to Speech Synthesizers*. 514–521. https://doi.org/10.1007/978-3-319-45510-5_59

Burileanu, C., Buzo, A., Petre, C. S., Ghelmez-Hanes, D., & Cucu, H. (2010). Romanian Spoken Language Resources and Annotation for Speaker Independent Spontaneous Speech Recognition. *2010 5th International Conference on Digital Telecommunications*, 7–10. https://doi.org/10.1109/icdt.2010.9

Cervantes, G., & Larocca, S. (2021). *TensorFlow Lite Extension to OpenNMT-tf Documentation*. https://doi.org/10.21236/ad1135718

Chen, Z., Droppo, J., Li, J., & Xiong, W. (2018). Progressive Joint Modeling in Unsupervised Single-Channel Overlapped Speech Recognition. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, *26*(1), 184–196. https://doi.org/10.1109/taslp.2017.2765834

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2019). Unsupervised Cross-Lingual Representation Learning at Scale. *ArXiv:1911.02116*. https://doi.org/10.48550/arXiv.1911.02116

Continuumlabs2024. (n.d.). *Continuumlabs*. https://training.continuumlabs.ai/data/datasets/what-is-perplexity

Diwan, A., Vaideeswaran, R., Shah, S., Singh, A., Raghavan, S., Khare, S., Unni, V., Vyas, S., Rajpuria, A., Yarra, C., Mittal, A., Ghosh, P. K., Jyothi, P., Bali, K., Seshadri, V., Sitaram, S., Bharadwaj, S., Nanavati, J., Nanavati, R., & Sankaranarayanan, K. (2021). MUCS 2021: Multilingual and Code-Switching ASR Challenges for Low Resource Indian Languages. *Interspeech 2021*, 2446–2450. https://doi.org/10.21437/interspeech.2021-1339

Github. (2024). *Github*. https://github.com/SakshiRathi77/hindiSpeechPro-Automatic-Speech-Recognization

Gupta, A., Chadha, H. S., Shah, P., Chimmwal, N., Dhuriya, A., Gaur, R., & Raghavan, V. (2021). CLSRIL-23: Cross-lingual speech representations for Indic languages. *ArXiv:2107.07402*. https://doi.org/10.48550/arXiv.2107.07402

Hindi, M. (2024). *Mera Hindi*. https://merahindi.com/category/hindi-pronunciation-rules/190

Hirayama, N., Yoshino, K., Itoyama, K., Mori, S., & Okuno, H. G. (2015). Automatic Speech Recognition for Mixed Dialect Utterances by Mixing Dialect Language Models. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, *23*(2), 373–382. https://doi.org/10.1109/taslp.2014.2387414

Huggingface. (2024a). *Huggingface*. https://huggingface.co/smcproject/Malwhisper-v1-medium

Huggingface. (2024b). *Huggingface*. https://huggingface.co/thennal/whisper-medium-ml.

Jinal, H. T., & Dipti B., S. (2016). Speech Recognition System Architecture for Gujarati Language. *International Journal of Computer Applications*, *138*(12), 28–31. https://doi.org/10.5120/ijca2016909049

Kim, S., & Seltzer, M. L. (2018). Towards Language-Universal End-to-End Speech Recognition. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4914–4918. https://doi.org/10.1109/icassp.2018.8462201

Klein, G., Kim, Y., Deng, Y., Senellart, J., & Rush, A. (2017). OpenNMT: Open-Source Toolkit for Neural Machine Translation. *Proceedings of ACL 2017, System Demonstrations*, 67–72. https://doi.org/10.18653/v1/p17-4012

Kumar, M. G., Kuriakose, J., Thyagachandran, A., Kumar A, Arun, Seth, A., Prasad, L. D., Jaiswal, S., Prakash, A., & Murthy, H. (2021). Dual Script E2E Framework for Multilingual and Code-Switching ASR. *ArXiv:2106.01400*. https://doi.org/10.48550/arXiv.2106.01400

Lee, S.-C., Wang, J.-F., & Chen, M.-H. (2018). Threshold-Based Noise Detection and Reduction for Automatic Speech Recognition System in Human-Robot Interactions. *Sensors*, *18*(7), 2068–2401. https://doi.org/10.3390/s18072068

Malek, J., Zdansky, J., & Cerva, P. (2017). Robust Automatic Recognition of Speech with background music. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5210–5214. https://doi.org/10.1109/icassp.2017.7953150

Mihajlik, P., Tuske, Z., Tarján, B., Németh, B., & Fegyó, T. (2010). Improved Recognition of Spontaneous Hungarian Speech—Morphological and Acoustic Modeling Techniques for a Less Resourced Task. *IEEE Transactions on Audio, Speech and Language Processing*, *18*(6), 1588–1600. https://doi.org/10.1109/tasl.2009.2038807

Nayar, V. R. P. (1980). *Svanavijnjaanam. Trivandrum, Kerala: State Institute of Languages*.

Open-Speech-EkStep. (2022). *Open-Speech-EkStep*. https://github.com/Open-Speech-EkStep/vakyansh-models#pretrained-asr-models

Prakash, A., Leela Thomas, A., Umesh, S., & A Murthy, H. (2019). Building Multilingual End-to-End Speech Synthesisers for Indian Languages. *10th ISCA Workshop on Speech Synthesis (SSW 10)*, 194–199. https://doi.org/10.21437/ssw.2019-35

Pratap, V., Sriram, A., Tomasello, P., Hannun, A., Liptchinsky, V., Synnaeve, G., & Collobert, R. (2020). Massively Multilingual ASR: 50 Languages, 1 Model, 1 billion Parameters. *Interspeech 2020*, 4751–4755. https://doi.org/10.21437/interspeech.2020-2831

Pradhan, A., Prakash, A., Aswin Shanmugam, S., Kasthuri, G. R., Krishnan, R., & Murthy, H. A. (2015). Building Speech Synthesis Systems for Indian Languages. *2015 21st National Conference on Communications (NCC)*, 1–6. https://doi.org/10.1109/ncc.2015.7084931

Prakash, A., & Murthy, H. A. (2020). Generic Indic Text-to-Speech Synthesisers with Rapid Adaptation in an End-to-End Framework. *Interspeech 2020*, 2962–2966. https://doi.org/10.21437/interspeech.2020-2663

Shahnawazuddin, S., Deepak K. T., Pradhan, G., & Sinha, R. (2017). Enhancing Noise and Pitch Robustness of Children's ASR. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5225–5229. https://doi.org/10.1109/icassp.2017.7953153

Toshniwal, S., Sainath, T. N., Weiss, R. J., Li, B., Moreno, P., Weinstein, E., & Rao, K. (2018). Multilingual Speech Recognition with a Single End-to-End Model. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4904–4908. https://doi.org/10.1109/icassp.2018.8461972